

# ADAPTIVE PRIMAL-DUAL HYBRID GRADIENT METHODS FOR SADDLE-POINT PROBLEMS

TOM GOLDSTEIN, ERNIE ESSER, RICHARD BARANIUK

**ABSTRACT.** The Primal-Dual hybrid gradient (PDHG) method is a powerful optimization scheme that breaks complex problems into simple sub-steps. Unfortunately, PDHG methods require the user to choose stepsize parameters, and the speed of convergence is highly sensitive to this choice. We introduce new adaptive PDHG schemes that automatically tune the stepsize parameters for fast convergence without user inputs. We prove rigorous convergence results for our methods, and identify the conditions required for convergence. We also develop practical implementations of adaptive schemes that formally satisfy the convergence requirements. Numerical experiments show that adaptive PDHG methods have advantages over non-adaptive implementations in terms of both efficiency and simplicity for the user.

## CONTENTS

1. Introduction	2
1.1. Notation	2
2. The Primal-Dual Hybrid Gradient Method	3
2.1. Optimality Conditions and Residuals	4
3. Common Saddle-Point Problems	5
3.1. Total-Variation Denoising	5
3.2. TVL1	6
3.3. Globally Convex Segmentation	6
3.4. Compressed Sensing / Single-Pixel Cameras	6
3.5. $\ell_\infty$ Minimization	7
3.6. Linear Programming	8
4. Residual Balancing	8
5. Adaptive Methods	9
5.1. Adaptive PDHG	9
5.2. Backtracking PDHG	10
6. Convergence Theory	12
6.1. Variational Inequality Formulation	12
6.2. Stepsize Conditions	12
6.3. Convergence of the Adaptive Method	13
6.4. Convergence of the Backtracking Method	14
6.5. Convergence Proof	14
7. Numerical Results	19
7.1. Experiments	20
7.2. Discussion	24
8. Conclusion	24
References	24

---

*Date:* May 3, 2013.

## 1. INTRODUCTION

This manuscript considers saddle-point problems of form

$$(1) \quad \min_{x \in X} \max_{y \in Y} f(x) + \langle Ax, y \rangle - g(y)$$

where  $f$  and  $g$  are convex functions,  $A \in \mathbb{R}^{M \times N}$  is a matrix, and  $X \subset \mathbb{R}^N$  and  $Y \subset \mathbb{R}^M$  are convex sets.

The formulation (1) is particularly appropriate for solving inverse problems involving the  $\ell_1$  norm. Such problems take the form

$$(2) \quad \min_x |Sx| + \frac{\mu}{2} \|Bx - f\|^2$$

where  $|\cdot|$  denotes the  $\ell_1$  norm,  $B$  and  $S$  are linear operators, and  $\|\cdot\|$  is the  $\ell_2$  norm. The formulation (2) is useful because it naturally enforces sparsity of  $Sx$ . Many different problems can be addressed by choosing different values for the sparsity transform  $S$ .

In the context of image processing,  $S$  is frequently the gradient operator. In this case the  $\ell_1$  term becomes  $|\nabla u|$ , the total variation semi-norm [1]. Problems of this form arise whenever an image is to be recovered from incomplete or noise-contaminated data. In this case,  $B$  is often a Gaussian blur matrix or a sub-sampled fast transform, such as a Fourier or Hadamard transform.

As we will see below, the problem (2) can be put in the “saddle-point” form (1) and can then be addressed using the techniques described below. Many other common minimization problems can also be put in this common form, including image segmentation, TVL1 minimization, and general linear programming.

In many problems of practical interest,  $f$  and  $g$  do not share common properties, making it difficult to derive numerical schemes for (1) that address both terms simultaneously. However, it frequently occurs in practice that efficient algorithms exist for minimizing  $f$  and  $g$  independently. In this case, the Primal-Dual Hybrid Gradient (PDHG) [2, 3] method is quite useful. This method removes the coupling between  $f$  and  $g$ , enabling each term to be addressed separately. Because it decouples  $f$  and  $g$ , the steps of PDHG can often be written explicitly, as opposed to other splitting methods that require expensive minimization sub-problems.

One of the primary difficulties with PDHG is that it relies on step-size parameters that must be carefully chosen by the user. The speed of the method depends heavily on the choice of these parameters, and there is often no intuitive way to choose them.

We will present practical adaptive schemes that optimize the PDHG parameters automatically as the problem is solved. Our new methods are not only much easier to use in practice, but also result in much faster convergence than constant-stepsizes schemes. After introducing the adaptive methods, we prove new theoretical results that guarantee convergence of PDHG under very general circumstances, including adaptive stepsizes.

**1.1. Notation.** Given two vectors  $u, v \in \mathbb{R}^N$ , we will denote their discrete inner product by  $\langle u, v \rangle = \sum_i u_i v_i$ . In some situations, we will also use the “dot product” notation,  $u \cdot v = \langle u, v \rangle$ .

The formulation (1) can be generalized to handle complex-valued vectors. In this case, we will consider the real part of the inner product. We use the notion  $\Re\{\cdot\}$  to denote the real part of a vector or scalar.

We will make use of a variety of norms, including the  $\ell_2$ -norm,  $\|u\| = \sqrt{\sum_i u_i^2}$ , and the  $\ell_1$ -norm,  $|u| = \sum_i |u_i|$ .

When  $M$  is a symmetric positive definite matrix, we define the  $M$ -norm by

$$\|u\|_M^2 = \langle Mu, u \rangle.$$

If  $M$  is indefinite, this does not define a proper norm. In this case, we will still write  $\|u\|_M^2$  to denote the quantity  $\langle Mu, u \rangle$ , even though this is an abuse of notation.

For a matrix  $M$ , we can also define the “operator norm”

$$\|M\|_{op} = \max_u \frac{\|Mu\|}{\|u\|}.$$

If  $M$  is symmetric, then the operator norm is the spectral radius of  $M$ , which we denote by  $\rho(M)$ .

We use the notation  $\partial f$  to denote the sub-differential (i.e., generalized derivative) of a function  $f$ .

Finally, we will use  $\chi_C$  to denote the characteristic function of a convex set  $C$ , which is defined as follows:

$$\chi_C(x) = \begin{cases} 0, & \text{if } x \in C \\ \infty, & \text{otherwise.} \end{cases}$$

## 2. THE PRIMAL-DUAL HYBRID GRADIENT METHOD

The PDHG scheme has its roots in the well-known Arrow-Hurwicz scheme, which was originally proposed in the field of economics and later refined for solving saddle point problems by Popov [4]. While the simple structure of the Arrow-Hurwicz method made it appealing, tight stepsize restrictions and poor performance make this method impractical for many problems.

Research in this direction was reinvigorated by the introduction of PDHG, which converges rapidly for a wide range of stepsizes. PDHG originally appeared in a technical report by Zhu and Chan [5]. It was later analyzed for convergence in [2, 3], and studied in the context of image segmentation in [6]. An extensive technical study of the method and its variants is given by He and Yuan [7].

The PDHG method for solving (1) only requires us to evaluate the *proximal* operators of  $f$  and  $g$ . These operators are given by

PDHG is listed in Algorithm 1. The algorithm can be interpreted as a forward-backward algorithm in both the primal and dual parameters. In steps (2-3), the method decreases the energy (1) in  $x$  by first taking a gradient descent step with respect to the inner product term in (1), and then taking a “backward” or proximal step for  $f$ . In steps (5-6), the energy (1) is increased by changing  $y$ . A gradient ascent step is taken with respect to the inner product term, and then a backward step is taken with respect to  $g$ .

---

### Algorithm 1 Basic PDHG

---

**Require:**  $x_0 \in \mathbb{R}^N$ ,  $y_0 \in \mathbb{R}^M$ ,  $\sigma_k, \tau_k > 0$

```

1: while Not Converged do
2:    $\hat{x}_{k+1} = x_k - \tau_k A^T y_k$ 
3:    $x_{k+1} = \arg \min_{x \in X} f(x) + \frac{1}{2\tau_k} \|x - \hat{x}_{k+1}\|^2$ 
4:    $\bar{x}_{k+1} = x_{k+1} + (x_{k+1} - x_k)$ 
5:    $\hat{y}_{k+1} = y_k + \sigma_k A \bar{x}_{k+1}$ 
6:    $y_{k+1} = \arg \min_{y \in Y} g(y) + \frac{1}{2\sigma_k} \|y - \hat{y}_{k+1}\|^2$ 
7: end while
```

---

Note that steps 3 and 6 of Algorithm 1 require minimizations. These minimization steps can be written in a more compact form using the *proximal* operators of  $f$  and  $g$ :

$$(3) \quad \begin{aligned} J_{\tau F}(\hat{x}) &= \arg \min_{x \in X} f(x) + \frac{1}{2\tau} \|x - \hat{x}\|^2 \\ J_{\sigma G}(\hat{y}) &= \arg \min_{y \in Y} g(y) + \frac{1}{2\sigma} \|y - \hat{y}\|^2. \end{aligned}$$

Algorithm 1 has been analyzed in the case of constant stepsizes,  $\tau_k = \tau$  and  $\sigma_k = \sigma$ . In particular, it is known to converge as long as  $\sigma\tau < \frac{1}{\rho(A^T A)}$  [2, 3, 7]. However, PDHG typically does not converge when non-constant stepsizes are used, even in the case that  $\sigma_k \tau_k < \frac{1}{\rho(A^T A)}$ .

In this article, we identify the specific stepsize conditions that guarantee convergence and propose practical adaptive methods that enforce these conditions.

**Remark.** Step 4 of Algorithm 1 is a prediction step of the form  $\bar{x}_{k+1} = x_{k+1} + \theta(x_{k+1} - x_k)$ , where  $\theta = 1$ . Note that PDHG methods have been analyzed for  $\theta \in [-1, 1]$ . Some authors have even suggested non-constant values of  $\theta$  as a means of accelerating the method [3]. However, it is known that the case  $\theta = 1$  tolerates the largest stepsizes (see [7]), and as a result  $\theta = 1$  has been used almost exclusively in applications.

**2.1. Optimality Conditions and Residuals.** Problem (1) can be written in its unconstrained form using the characteristic functions for the sets  $X$  and  $Y$ . We have

$$(4) \quad \min_{x \in \mathbb{R}^N} \max_{y \in \mathbb{R}^M} f(x) + \chi_X(x) + \langle Ax, y \rangle - g(y) - \chi_Y(y).$$

Let  $F = \partial\{f + \chi_X(x)\}$ , and  $G = \partial\{g + \chi_Y(y)\}$ <sup>1</sup>. From the equivalence between (1) and (4), we see that the optimality conditions for (1) are

$$(5) \quad 0 \in F(x^*) + A^T y^*$$

$$(6) \quad 0 \in G(y^*) - A(x^*).$$

The optimality conditions (5,6) state that the derivative with respect to both the primal and dual variables must be zero. This motivates us to define the primal and dual residuals

$$(7) \quad \begin{aligned} P(x, y) &= F(x) + A^T y \\ D(x, y) &= G(y) - Ax. \end{aligned}$$

We can measure convergence of the algorithm by tracking the size of these residuals. Note that the residuals are in general multi-valued (because they depend on the subdifferentials of  $f$  and  $g$ , as well as the characteristic functions of  $X$  and  $Y$ ). We can obtain explicit formulas for these residuals by observing the optimality conditions for step 2 and 4 of Algorithm 1:

$$(8) \quad 0 \in F(x_{k+1}) + A^T y_k + \frac{1}{\tau_k} (x_{k+1} - x_k)$$

$$(9) \quad 0 \in G(y_{k+1}) - A(2x_{k+1} - x_k) + \frac{1}{\sigma_k} (y_{k+1} - y_k).$$

---

<sup>1</sup>Note that  $\partial\{f + \chi_X(x)\} = \partial f + \partial\chi_X(x)$  precisely when the resolvent minimizations (3) are feasible (see Rockafellar [8] Theorem 23.8), and in this case the optimality conditions (4) admit a solution. We will assume for the remainder of this article that the problem (4) is feasible.

Manipulating these optimality conditions yields

$$(10) \quad P_{k+1} = \frac{1}{\tau_k}(x_k - x_{k+1}) - A^T(y_k - y_{k+1}) \in F(x_{k+1}) + A^T y_{k+1}$$

$$(11) \quad D_{k+1} = \frac{1}{\sigma_k}(y_k - y_{k+1}) - A(x_k - x_{k+1}) \in G(y_{k+1}) - Ax_{k+1}.$$

Formulas (10) and (11) define a sequence of primal and dual residual vectors such that  $P_k \in P(x_k, y_k)$  and  $D_k \in D(x_k, y_k)$ .

We say that Algorithm 1 converges if

$$\lim_{k \rightarrow \infty} \|P_k\|^2 + \|D_k\|^2 = 0.$$

Note that studying the residual convergence of Algorithm 1 is more general than studying the convergence of subsequences of iterates, because the solution to (1) need not be unique.

### 3. COMMON SADDLE-POINT PROBLEMS

Many common variational problems have saddle-point formulations that are efficiently solved using PDHG. While the applications of saddle-point problems are vast, we focus here on several simple problems that commonly appear in image processing.

**3.1. Total-Variation Denoising.** A ubiquitous problem in image processing is minimizing the Rudin-Osher-Fatemi (ROF) denoising model [1]:

$$(12) \quad \min_x |\nabla x| + \frac{\mu}{2} \|x - f\|^2.$$

The energy (12) contains two terms. The  $\ell_2$  term on the right minimizes the squared error between the recovered image and the noise-contaminated measurements,  $f$ . The TV term,  $|\nabla x|$ , enforces that the recovered image be smooth in the sense that its gradient has sparse support.

The problem (12) can be formulated as a saddle-point problem by writing the TV term as a maximization over the “dual” variable  $y \in \mathbb{R}^{2 \times N}$ , where the image  $x \in \mathbb{R}^N$  has  $N$  pixels:

$$(13) \quad TV(x) = |\nabla x| = \max_{\|y\|_\infty \leq 1} y \cdot \nabla x.$$

Note that the maximization is taken over the set

$$C_\infty = \{y \in \mathbb{R}^{2 \times N} \mid y_{1,i}^2 + y_{2,i}^2 \leq 1\}.$$

The TV-regularized problem (12) can then be written in its primal-dual form

$$(14) \quad \max_{y \in C_\infty} \min_x \frac{\mu}{2} \|x - f\|^2 + y \cdot \nabla x$$

which is clearly a saddle-point problem in the form (1) with  $f = \frac{\mu}{2} \|x - f\|^2$ ,  $A = \nabla$ ,  $g = 0$ ,  $X = \mathbb{R}^N$ , and  $Y = C_\infty$ .

To apply Algorithm 1, we need efficient solutions to the sub-problems in steps 3 and 6, which can be written

$$(15) \quad J_{\tau F}(\hat{x}) = \arg \min_x \frac{\mu}{2} \|x - f\|^2 + \frac{1}{2\tau} \|x - \hat{x}\|^2 = \frac{\tau}{\tau\mu + 1} (\mu f + \frac{1}{\tau} \hat{x})$$

$$(16) \quad J_{\sigma G}(\hat{y}) = \arg \min_{y \in C_\infty} \frac{1}{2\sigma} \|y - \hat{y}\|^2 = \left( \frac{y_i}{\max\{y_i, 1\}} \right)_{i=1}^M.$$

**3.2. TVL1.** Another common denoising model is TVL1 [9]. This corresponds to minimizing the energy

$$(17) \quad \min_x |\nabla x| + \mu |x - f|$$

which is similar to (12), except that the data term relies on the  $\ell_1$  norm instead of  $\ell_2$ . This model is very effective for problems involving “heavy-tailed” noise, such as shot noise or salt-and-pepper noise.

To put the energy (17) into the form (1), we write the data term in its variational form

$$\mu |x - f| = \max_{y \in C_\mu} \langle y, x - f \rangle$$

where  $C_\mu = \{y \in R^N \mid |y_i| \leq \mu\}$ .

The energy (17) can then be minimized using the formulation

$$(18) \quad \max_{y_1 \in C_\infty, y_2 \in C_\mu} \min_x y_1 \cdot \nabla x + \langle y_2, x \rangle - \langle y_2, f \rangle$$

which is of the form (1).

**3.3. Globally Convex Segmentation.** Image segmentation is the task of grouping pixels together by intensity and spatial location. One of the simplest models for image segmentation is the globally convex segmentation model of Chan, Esedoglu, and Nikolova (CEN) [10, 11]. Given an image  $f$  and two real numbers  $c_1$  and  $c_2$  we wish to partition the pixels into two groups depending on whether their intensity lies closer to  $c_1$  or  $c_2$ . Simultaneously, we want the regions to have smooth boundaries.

The CEN segmentation model has the variational form

$$(19) \quad \min_{0 \leq x \leq 1} |\nabla x| + \langle x, l \rangle$$

where  $l_i = (f_i - c_1)^2 - (f_i - c_2)^2$ . The inner-product term of the right forces the entries in  $x$  toward either 1 or 0, depending on whether the corresponding pixel intensity in  $f$  is closer to  $c_1$  or  $c_2$ . At the same time, the TV term enforces that the boundary is smooth.

Using the identity (13), we can write the model (19) as

$$(20) \quad \max_{y \in C_\infty} \min_{x \in [0,1]} y \cdot \nabla x + \langle x, l \rangle.$$

We can then apply Algorithm 1, where step 3 takes the form

$$J_{\tau F}(x) = \max\{0, \min\{x, 1\}\}$$

and step 6 is given by (16).

Generalizations of (19) to multi-label segmentations and general convex regularizers have been presented in [12, 13, 14, 15]. Many of these models result in minimizations similar to (19), and PDHG methods have become a popular scheme for solving these models.

**3.4. Compressed Sensing / Single-Pixel Cameras.** In compressed sensing, one is interested in reconstructing images from incomplete measurements taken in an orthogonal transform domain (such as the Fourier or Hadamard domain). It has been shown that high-resolution images can be obtained from incomplete data as long as the image has a sparse representation [16, 17], for example a sparse gradient.

The Single Pixel Camera (SPC) is an imaging modality that leverages compressed sensing to reconstruct images using a small number of measurements [18]. Rather than measuring in the pixel domain like conventional cameras, SPC’s measure a subset of the Hadamard transform coefficients of an image.

To reconstruct images, SPC's rely on variational problems of the form

$$(21) \quad |\nabla x| + \frac{\mu}{2} \|RHx - b\|^2$$

where  $b$  contains the measured transform coefficients of the image, and  $H$  is an orthogonal transform matrix (such as the Hadamard transform). Here,  $R$  is a diagonal “row selector” matrix, with diagonal elements equal to 1 if the corresponding Hadamard row has been measured, and 0 if it has not.

This problem can be put into the saddle-point form

$$\max_{y \in C_\infty} \min_x \frac{\mu}{2} \|RHx - b\|^2 + y \cdot \nabla x$$

and then solved using PDHG.

The solution to step 6 of Algorithm 1 is given by (16). Step 3 requires we solve

$$\min_x \frac{\mu}{2} \|RHx - b\|^2 + \frac{1}{2\tau} \|x - \hat{x}\|^2.$$

Because  $H$  is orthogonal, we can write the solution to this problem explicitly as

$$x = H^T \left( \mu R + \frac{1}{\tau} I \right)^{-1} H \left( \mu H^T R b + \frac{1}{\tau} \hat{x} \right).$$

Note that for compressed sensing problems of the form (21), PDHG has the advantage that all steps of the method can be written explicitly, and no expensive minimization sub-steps are needed.

**3.5.  $\ell_\infty$  Minimization.** In wireless communications, one is often interested in finding signal representations with low peak-to-average power ratio (PAPR) [19, 20]. Given a signal  $z$  with high dynamic range, we can obtain a more efficient representation by solving

$$(22) \quad \min_x \|x\|_\infty \text{ subject to } \|Dx - z\| \leq \epsilon$$

where  $x$  is the new representation,  $D$  is the representation basis, and  $\epsilon$  is the desired level of representation accuracy [20]. Minimizations of the form (22) also arise in numerous other applications, including robotics [21] and nearest neighbor search [22].

The problem (22) can be written in the constrained form

$$\min_{x_1 \in \mathbb{R}^N, x_2 \in C_\epsilon} \|x_1\|_\infty \text{ subject to } x_2 = Dx_1 - z$$

where  $C_\epsilon = \{z \mid \|z\| < \epsilon\}$ . When we introduce Lagrange multipliers  $y$  to enforce the equality constraint, we arrive at the saddle-point formulation

$$\max_{y \in \mathbb{R}^M} \min_{x_1 \in \mathbb{R}^N, x_2 \in C_\epsilon} \|x_1\|_\infty + \langle y, Dx_1 - x_2 - z \rangle$$

which is of the form (1).

It often happens that  $D$  is a complex-valued frame (such as a sub-sampled Fourier matrix). In this case,  $x$  and  $y$  can take on complex values. The appropriate saddle-point problem is then

$$\max_{y \in \mathbb{R}^M} \min_{x_1 \in \mathbb{R}^N, x_2 \in C_\epsilon} \|x_1\|_\infty + \Re\{\langle y, Dx_1 - x_2 - z \rangle\}$$

and we can apply PDHG just as we did for real-valued problems provided we are careful to use the Hermitian definition of the matrix transpose (see the remark at the end of Section 5.2).

To apply PDHG to this problem, we need to evaluate the proximal minimization

$$\min_x \|x\|_\infty + \frac{1}{2t} \|x - \hat{x}\|^2$$

which can be computed efficiently using a sorting algorithm [23].

**3.6. Linear Programming.** Suppose we wish to solve a large-scale linear program in the canonical form

$$(23) \quad \begin{aligned} \min_x \quad & \langle c, x \rangle \\ \text{subject to} \quad & Ax \leq b, x \geq 0 \end{aligned}$$

where  $c$  and  $b$  are vectors, and  $A$  is a matrix [24]. This problem can be written in the primal-dual form

$$\max_{y \geq 0} \min_{x \geq 0} \langle c, x \rangle + \langle y, Ax - b \rangle$$

where  $y$  can be interpreted as Lagrange multipliers enforcing the condition  $Ax \leq b$  [24].

Steps 3 and 6 of Algorithm 1 are now simply

$$\begin{aligned} J_{\tau F}(\hat{x}) &= \max\{\hat{x} - \tau c, 0\} \\ J_{\sigma G}(\hat{y}) &= \max\{\hat{y} - \sigma b, 0\}. \end{aligned}$$

The PDHG solution to a linear program is advantageous for extremely large problems for which conventional simplex and interior-point solvers are intractable. Primal-dual solvers for (23) have been studied in [25], and compared to conventional solvers.

#### 4. RESIDUAL BALANCING

When choosing the stepsize for PDHG, there is a tradeoff between the primal and dual residuals. Choosing a large value of  $\tau_k$  creates a very powerful minimization step in the primal variables and a slow maximization step in the dual variables, resulting in very small primal residuals at the cost of large dual residuals. Choosing  $\tau_k$  to be small, on the other hand, results in small dual residuals at the cost of large primal errors.

Ideally, one would like to choose stepsizes so that the larger of  $P_k$  and  $D_k$  is as small as possible. If we assume the primal/dual residuals decrease/increase monotonically with  $\tau_k$ , then  $\max\{P_k, D_k\}$  is minimized when both residuals are equal in magnitude. This suggests that  $\tau_k$  be chosen to “balance” the primal and dual residual – i.e., the primal and dual residuals should be roughly the same size, up to some scaling to account for units. This principle has been suggested for other iterative methods (see [26, 27]).

The particular residual balancing principle we suggest is to enforce

$$(24) \quad |P_k| \approx s|D_k|$$

where  $|\cdot|$  denotes the  $\ell_1$  norm, and  $s$  is a scaling parameter. Residual balancing methods work by “tuning” parameters after each iteration to approximately maintain this equality. If the primal residual grows disproportionately large,  $\tau_k$  is increased and  $\sigma_k$  is decreased (or vice-versa).

In (24) we use the  $\ell_1$ -norm to measure the size of the residuals. In principle,  $\ell_1$  could easily be replaced with the  $\ell_2$  norm, or any other norm. We have found that the  $\ell_1$  norm performs somewhat better than the  $\ell_2$  norm for some problems, because it is less sensitive to outliers that may dominate the residual.

Note the proportionality in (24) depends on a constant  $s$ . This is to account for the effect of scaling the inputs (i.e., changing units). For example, in the TVL1 model (17) the input image  $f$  could be scaled with pixel intensities in  $[0, 1]$  or  $[0, 255]$ . As long as the primal step



sizes used with the latter scaling are 255 times that of the former (and the dual step sizes are 255 times smaller) both problems produce similar sequences of iterates. However, in the  $[0, 255]$  case the dual residuals are 255 times larger than in the  $[0, 1]$  case while the primal residuals are the same.

For image processing problems, we recommend using the  $[0, 255]$  scaling with  $s = 1$ , or equivalently the  $[0, 1]$  scaling with  $s = 255$ . This scaling enforces that the saddle point objective (1) is nearly maximized with respect to  $y$  and that the saddle point term (13) is a good approximation of the total variation semi-norm.

## 5. ADAPTIVE METHODS

In this section, we develop new the adaptive PDHG methods. The first method assumes we have a bound on  $\rho(A^T A)$ . In this case, we can enforce the stability condition

$$(25) \quad \tau_k \sigma_k < L < \rho(A^T A)^{-1}.$$

This is the same stability condition that guarantees convergence in the non-adaptive case. In the adaptive case, this condition alone is not sufficient for convergence but leads to relatively simple methods.

The second method does not require any knowledge of  $\rho(A^T A)$ . Rather, we use a backtracking scheme to enforce that the stepsizes are small enough to guarantee convergence. This is similar to the Armijo-type backtracking line searches that are used in conventional convex optimization. In addition to requiring no knowledge of  $\rho(A^T A)$ , the backtracking scheme has the advantage that it can use relatively long steps that violate the stability condition (25). Especially for problems with highly sparse solutions, this can lead to faster convergence than methods that enforce (25).

Both of these methods have strong convergence guarantees. In particular, we show in Section 6 that both methods converge in the sense that the norm of the residuals goes to zero.

**5.1. Adaptive PDHG.** The first adaptive method is listed in Algorithm 2. The loop in Algorithm 2 begins by performing a standard PDHG step using the current stepsizes. In steps 4 and 5, we compute the primal and dual residuals and store their  $\ell_1$  norms in  $p_k$  and  $d_k$ . If the primal residual is sufficiently large compared to the dual residual then the primal stepsize is increased by a factor of  $(1 - \alpha_k)^{-1}$ , and the dual stepsize is decreased by a factor of  $(1 - \alpha_k)$ . If the primal residual is somewhat smaller than the dual residual then the primal stepsize is decreased and the dual stepsize is increased. If both residuals are comparable in size, then the stepsizes remain the same on the next iteration.

The parameter  $\Delta > 1$  is used to compare the sizes of the primal and dual residuals. The stepsizes are only updated if the residuals differ by a factor greater than  $\Delta$ .

The sequence  $\{\alpha_k\}$  controls the adaptivity level of the method. We start with some  $\alpha_0 \in (0, 1)$ . Every time we choose to update the stepsize parameters, we define  $\alpha_{k+1} = \eta \alpha_k$  for some  $\eta < 1$ . In this way, the adaptivity decreases over time. We will show in Section 6 that this definition of  $\{\alpha_k\}$  is needed to guarantee convergence of the method.

**Remark.** The adaptive scheme requires several arbitrary constants as inputs. These are fairly easy to choose in practice. A fairly robust choice is  $a_0 = 0.5$ ,  $\Delta = 1.5$ ,  $\eta = 0.95$ , and  $\tau_0 = \sigma_0 = 1/\sqrt{\rho(A^T A)}$ . However, these parameters can certainly be tuned for various applications. As stated above, we recommend choosing  $s = 1$  for imaging problems when pixels lie in  $[0, 255]$ , and  $s = 255$  when pixels lie in  $[0, 1]$ .

**Algorithm 2** Adaptive PDHG

---

**Require:**  $x_0 \in \mathbb{R}^N, y_0 \in \mathbb{R}^M, \sigma_0 \tau_0 < \rho(A^T A)^{-1}, (\alpha_0, \eta) \in (0, 1)^2, \Delta > 1, s > 0$

```

1: while  $p_k, d_k > \text{tolerance}$  do
2:    $x_{k+1} = J_{\tau_k F}(x_k - \tau_k A^T y_k)$                                  $\triangleright$  Begin with normal PDHG
3:    $y_{k+1} = J_{\sigma_k G}(y_k + \sigma_k A(2x_{k+1} - x_k))$ 
4:    $p_{k+1} = |(x_k - x_{k+1})/\tau_k - A^T(y_k - y_{k+1})|$                  $\triangleright$  Compute primal residual
5:    $d_{k+1} = |(y_k - y_{k+1})/\sigma_k - A(x_k - x_{k+1})|$                  $\triangleright$  Compute dual residual
6:   if  $p_{k+1} > s d_{k+1} \Delta$  then                                     $\triangleright$  If primal residual is large...
7:      $\tau_{k+1} = \tau_k / (1 - \alpha_k)$                                      $\triangleright$  Increase primal stepsize
8:      $\sigma_{k+1} = \sigma_k (1 - \alpha_k)$                                  $\triangleright$  Decrease dual stepsize
9:      $\alpha_{k+1} = \alpha_k \eta$                                              $\triangleright$  Decrease adaptivity level
10:  end if
11:  if  $p_{k+1} < s d_{k+1} / \Delta$  then                                     $\triangleright$  If dual residual is large...
12:     $\tau_{k+1} = \tau_k (1 - \alpha_k)$                                      $\triangleright$  Decrease primal stepsize
13:     $\sigma_{k+1} = \sigma_k / (1 - \alpha_k)$                                  $\triangleright$  Increase dual stepsize
14:     $\alpha_{k+1} = \alpha_k \eta$                                              $\triangleright$  Decrease adaptivity level
15:  end if
16:  if  $s d_{k+1} / \Delta \leq p_{k+1} \leq s d_{k+1} \Delta$  then                 $\triangleright$  If residuals are similar...
17:     $\tau_{k+1} = \tau_k$                                                  $\triangleright$  Leave primal step the same
18:     $\sigma_{k+1} = \sigma_k$                                              $\triangleright$  Leave dual step the same
19:     $\alpha_{k+1} = \alpha_k$                                              $\triangleright$  Leave adaptivity level the same
20:  end if
21: end while

```

---

**Remark.** The computation of the residuals in steps 4 and 5 of Algorithm 2 requires multiplications by  $A^T$  and  $A$ , which seems at first to increase the cost of each iteration. Note however that the values of  $A^T y_k$  and  $A x_k$  are already computed in the other steps of the algorithm. If we simply note that  $A^T(y_k - y_{k+1}) = A^T y_k - A^T y_{k+1}$  and  $A(x_k - x_{k+1}) = A x_k - A x_{k+1}$ , then we can evaluate the residuals in steps 4 and 5 without any additional multiplications by  $A$  or  $A^T$ .

**5.2. Backtracking PDHG.** In situations where bounds on  $\rho(A^T A)$  are unavailable, or when the stability condition (25) is overly conservative, the backtracking method presented here becomes valuable. PDHG with backtracking is presented in Algorithm 3.

The backtracking scheme is similar to the simple adaptive method (Algorithm 2) with one key difference. We will see in Section 6 that convergence is guaranteed if the following “backtracking” condition holds at each step:

$$(26) \quad b_k = \frac{2\tau_k \sigma_k \langle A(x_{k+1} - x_k), y_{k+1} - y_k \rangle}{\gamma \sigma_k \|x_{k+1} - x_k\|^2 + \gamma \tau_k \|y_{k+1} - y_k\|^2} \leq 1.$$

where  $\gamma \in (0, 1)$  is a constant. If this inequality does *not* hold, then the stepsizes are too large. In this case we decrease the stepsizes by a factor of  $b_k \beta$ , where  $\beta \in (0, 1)$  is a constant. Reasons for this particular update choice are elaborated in the Section 6.4.

We will show in Section 6 that this backtracking step can only be activated a finite number of times. Furthermore, enforcing the condition (26) at each step is sufficient to guarantee convergence, provided that one of the spaces  $X$  or  $Y$  is bounded.

We recommend choosing  $\gamma = 0.75$  and  $\beta = 0.95$ , although the method is convergent for any  $\gamma, \beta \in (0, 1)$ .

**Algorithm 3** Backtracking PDHG

---

**Require:**  $x_0 \in \mathbb{R}^N, y_0 \in \mathbb{R}^M, (\tau_0, \sigma_0, s) > 0, \Delta > 1, (\eta, \alpha_0, \beta, \gamma) \in (0, 1)^4$

1: <b>while</b> $p_k, d_k > \text{tolerance}$ <b>do</b>	
2: $x_{k+1} = J_{\tau_k F}(x_k - \tau_k A^T y_k)$	▷ Begin with normal PDHG
3: $y_{k+1} = J_{\sigma_k G}(y_k + \sigma_k A(2x_{k+1} - x_k))$	
4: $p_{k+1} =  (x_k - x_{k+1})/\tau_k - A^T(y_k - y_{k+1}) $	▷ Compute primal residual
5: $d_{k+1} =  (y_k - y_{k+1})/\sigma_k - A(x_k - x_{k+1}) $	▷ Compute dual residual
6: $b_k = \frac{2\tau_k \sigma_k \langle A(x_{k+1} - x_k), y_{k+1} - y_k \rangle}{\gamma \sigma_k \ x_{k+1} - x_k\ ^2 + \gamma \tau_k \ y_{k+1} - y_k\ ^2}$	▷ Compute stability condition
7: <b>if</b> $b_k > 1$	▷ If stability condition fails...
8: $\tau_{k+1} = \beta \tau_k / b_k, \sigma_{k+1} = \beta \sigma_k / b_k$	▷ Decrease stepsizes
9: $x_{k+1} = x_k, y_{k+1} = y_k$	▷ Keep old iterates
10: $\alpha_{k+1} = \alpha_0$	▷ Reset adaptivity parameter
11: <b>else if</b> $p_{k+1} > s d_{k+1} \Delta$	▷ If primal residual is large...
12: $\tau_{k+1} = \tau_k / (1 - \alpha_k)$	▷ Increase primal stepsize
13: $\sigma_{k+1} = \sigma_k (1 - \alpha_k)$	▷ Decrease dual stepsize
14: $\alpha_{k+1} = \alpha_k \eta$	▷ Decrease adaptivity level
15: <b>else if</b> $p_{k+1} < s d_{k+1} / \Delta$	▷ If dual residual is large...
16: $\tau_{k+1} = \tau_k (1 - \alpha_k)$	▷ Decrease primal stepsize
17: $\sigma_{k+1} = \sigma_k / (1 - \alpha_k)$	▷ Increase dual stepsize
18: $\alpha_{k+1} = \alpha_k \eta$	▷ Decrease adaptivity level
19: <b>else if</b> $s d_{k+1} / \Delta \leq p_{k+1} \leq s d_{k+1} \Delta$	▷ If residuals are similar...
20: $\tau_{k+1} = \tau_k$	▷ Leave primal step the same
21: $\sigma_{k+1} = \sigma_k$	▷ Leave dual step the same
22: $\alpha_{k+1} = \alpha_k$	▷ Leave adaptivity level the same
23: <b>end</b>	
24: <b>end while</b>	

---

The initial stepsizes  $\tau_0$  and  $\sigma_0$  should be chosen so that  $\tau_0 \sigma_0$  is somewhat larger than  $\rho(A^T A)^{-1}$ . In the numerical experiments below, we choose

$$(27) \quad \tau_0 = \sigma_0 = \sqrt{\frac{2\|x_r\|}{\|A^T A x_r\|}}$$

where  $x_r$  is a random Gaussian distributed vector. Note that the ratio  $\frac{\|A^T A x_r\|}{\|x_r\|}$  is guaranteed to be less than  $\rho(A^T A)$ , and thus  $\tau_0 \sigma_0 > \rho(A^T A)^{-1}$ . The factor of 2 is included to account for problems for which the method is stable with large steps.

**Remark.** Note that our convergence theorems require either  $X$  or  $Y$  to be bounded. This requirement is indeed satisfied by most of the common saddle-point problems described in Section 3. However, we have found empirically that the scheme is quite stable even for unbounded  $X$  and  $Y$ . For the linear programming example, neither space is bounded. Nonetheless, the backtracking method converges reliably in practice.

**Remark.** In the case of complex-valued problems, both  $x$  and  $y$  may take on complex values, and the relevant saddle-point problem is

$$\min_{x \in X} \max_{y \in Y} f(x) + \Re\{\langle Ax, y \rangle\} - g(y)$$

where  $\Re\{\cdot\}$  denotes the real part of a vector. In this case the numerator of  $b_k$  may have an imaginary component. The algorithm is still convergent as long as we replace  $b_k$  with its real part and use the definition

$$(28) \quad b_k = \frac{\Re\{2\tau_k\sigma_k\langle A(x_{k+1} - x_k), y_{k+1} - y_k \rangle\}}{\gamma\sigma_k\|x_{k+1} - x_k\|^2 + \gamma\tau_k\|y_{k+1} - y_k\|^2}$$

in Algorithm 3.

## 6. CONVERGENCE THEORY

**6.1. Variational Inequality Formulation.** For notational simplicity, we define the vector quantities

$$(29) \quad u_k = \begin{pmatrix} x_k \\ y_k \end{pmatrix}, \quad R(u) = \begin{pmatrix} P(x, y) \\ D(x, y) \end{pmatrix},$$

and the matrices

$$(30) \quad M_k = \begin{pmatrix} \tau_k^{-1}I & -A^T \\ -A & \sigma_k^{-1}I \end{pmatrix}, \quad H_k = \begin{pmatrix} \tau_k^{-1}I & 0 \\ 0 & \sigma_k^{-1}I \end{pmatrix}.$$

This notation was first suggested to simplify PDHG by He and Yuan [7].

Following [7], it is possible to formulate each step of PDHG as a solution to the variational inequality

$$(31) \quad 0 \in R(u_{k+1}) + M_k(u_{k+1} - u_k).$$

Also, the optimality conditions (5) and (6) can be written succinctly as

$$(32) \quad 0 \in R(u^*).$$

Note that  $R$  is monotone (see [28]), meaning that

$$\langle u - \hat{u}, R(u) - R(\hat{u}) \rangle \geq 0, \quad \forall u, \hat{u}.$$

**6.2. Stepsize Conditions.** Here, we state the conditions that guarantee convergence of adaptive PDHG. We begin by defining the following constants:

$$(33) \quad \begin{aligned} \delta_k &= \min \left\{ \frac{\tau_{k+1}}{\tau_k}, \frac{\sigma_{k+1}}{\sigma_k}, 1 \right\} \\ \phi_k &= 1 - \delta_k \geq 0. \end{aligned}$$

These constants quantify how much the stepsizes decrease with each iteration.

Consider the following conditions on the step-sizes in Algorithm 1:

**Convergence Conditions for Adaptive PDHG**

Algorithm 1 converges if the following three requirements hold:

**A:** The sequences  $\{\tau_k\}$  and  $\{\sigma_k\}$  are bounded.

**B:** The sequence  $\{\phi_k\}$  is summable, i.e.

$$\sum_{k \geq 0} \phi_k < \infty.$$

**C:** One of the following two conditions is met:

**C1:** There is a constant  $L$  such that for all  $k > 0$

$$\tau_k \sigma_k < L < \rho(A^T A)^{-1}.$$

**C2:** Either  $X$  or  $Y$  is bounded, and there is a constant  $c \in (0, 1)$  such that for all  $k > 0$

$$\|u_{k+1} - u_k\|_{M_k}^2 \geq c \|u_{k+1} - u_k\|_{H_k}^2.$$

Two conditions must always hold to guarantee convergence: **A** The stepsizes must remain bounded, and **B** the sequence  $\{\phi_i\}$  must be summable. Together, these conditions ensure that the steps sizes do not oscillate too wildly as  $k$  gets large.

In addition, either **C1** or **C2** must hold. When we know the spectral radius of  $A^T A$ , we can use condition **C1** to enforce that the method is stable. When we have no knowledge of  $\rho(A^T A)$ , or when we want to take large stepsizes, the backtracking condition **C2** can also be used to enforce stability. We will see that for small enough stepsizes, this backtracking condition can always be satisfied.

Note that condition **C1** is slightly stronger than condition **C2**. The advantage of condition **C1** is that it results in somewhat simpler methods because backtracking does not need to be used. However when backtracking is used to enforce condition **C2**, we can take larger steps that violate condition **C1**.

In the following sub-sections, we explain how Algorithms 2 and 3 explicitly satisfy conditions **A**, **B** and **C**, and are therefore guaranteed to converge. In Section 6.5, we prove convergence of the general Algorithm 1 under assumptions **A**, **B** and **C**.

**6.3. Convergence of the Adaptive Method.** Algorithm 2 is a practical implementation of adaptive PDHG that satisfies conditions **A**, **B**, and **C1**. An examination of Algorithm 2 reveals that, depending on the balance between the residuals, there are three possible stepsize updates. Regardless of which update occurs, the product  $\tau_k \sigma_k$  remains unchanged at each iteration, and so  $\tau_k \sigma_k = \tau_0 \sigma_0 = L < \rho(A^T A)^{-1}$ . It follows that condition **A** and **C1** are satisfied.

Also, because  $0 < \alpha_k < 1$ , we have

$$\phi_k = 1 - \min\left\{\frac{\tau_{k+1}}{\tau_k}, \frac{\sigma_{k+1}}{\sigma_k}, 1\right\} = \begin{cases} \alpha_k, & \text{if the steps are updated} \\ 1, & \text{otherwise.} \end{cases}$$

Note that every time we update the stepsizes, we update  $\alpha_{k+1} = \eta \alpha_k$  where  $\eta < 1$ . Thus the non-zero entries in the sequence  $\{\phi_k\}$  form a decreasing geometric sequence. It follows that the summation condition **B** is satisfied.

Because Algorithm 2 satisfies conditions **A**, **B** and **C**, its convergence is guaranteed by the theory presented in Section 6.5.

**6.4. Convergence of the Backtracking Method.** Algorithm 3 uses the same adaptive stepsize updates as Algorithm 2. As we shall prove in Section 6.5, the backtracking step is triggered only a finite number of times. Thus, the backtracking method satisfies conditions **A** and **B**.

We can expand condition **C2** using the definition of  $\|\cdot\|_{M_k}$  and  $\|\cdot\|_{H_k}$  to obtain

$$\begin{aligned} & \frac{1}{\tau_k} \|x_{k+1} - x_k\|^2 - 2\langle A(x_{k+1} - x_k), y_{k+1} - y_k \rangle + \frac{1}{\sigma_k} \|y_{k+1} - y_k\|^2 \\ & > \frac{c}{\tau_k} \|x_{k+1} - x_k\|^2 + \frac{c}{\sigma_k} \|y_{k+1} - y_k\|^2. \end{aligned}$$

If we combine like terms and let  $\gamma = 1 - c$ , then this condition is equivalent to

$$(34) \quad \frac{\gamma}{\tau_k} \|x_{k+1} - x_k\|^2 + \frac{\gamma}{\sigma_k} \|y_{k+1} - y_k\|^2 > 2\langle A(x_{k+1} - x_k), y_{k+1} - y_k \rangle.$$

If we note that the left side of (34) is non-negative, then we can easily see that this condition is equivalent to (26). Thus, the backtracking conditions (26) explicitly enforce condition **C2**, and the convergence of Algorithm 3 is guaranteed by the analysis below.

We now address the form of the backtracking update in line 8 of Algorithm 3. A simple Armijo-type backtracking scheme would simply choose  $\tau_{k+1} = \xi_k \tau_k$  and  $\sigma_{k+1} = \xi_k \sigma_k$ , where  $\xi < 1$ , every time that (34) is violated. However if our initial guess for  $L$  is very large, then this backtracking scheme could be very slow. Rather, we wish to predict the value of  $\xi$  that makes (34) hold. To do this, we assume that the values of  $\|x_{k+1} - x_k\|$  and  $\|y_{k+1} - y_k\|$  decrease linearly with  $\xi$ . Under this assumption, the value of  $\xi$  that makes (34) into an equality is  $\xi_k = b_k^{-1}$ . In order to guarantee that the stepsizes can become arbitrarily small, we make the slightly more conservative choice of  $\xi_k = \beta/b_k$  for  $\beta < 1$ .

**6.5. Convergence Proof.** In this section, we prove convergence of Algorithm 1 under assumptions **A**, **B** and **C**.

The overall strategy of this proof is to show that the PDHG method satisfies a *Fejér inequality*, i.e., an inequality stating that the iterates move toward the solution set on every iteration. We derive a simple Fejér inequality in Lemma 2. This lemma states that  $u_{k+1}$  always lies closer to the solution set than  $u_k$  as measured in the  $M_k$  norm. Normally, this would be enough to prove convergence (see [28], chapter 5). However, in our case the proof is not so straight-forward for several reasons. First, the  $M_k$ -norm changes every time we update the stepsizes. Second, when the backtracking condition **C2** is used,  $M_k$  may be indefinite, in which case  $\|\cdot\|_{M_k}$  is not a proper norm, and it can even assume negative values.

Nonetheless, we can still prove convergence. We begin by proving that **C1** guarantees that  $M_k$  is positive definite. We then derive a Fejér inequality in Lemma 2. In cases when  $M_k$  is indefinite, Lemmas 3 and 4 show that terms involving the  $M_k$ -norm remain uniformly bounded over all iterations. These bounding conditions will be strong enough to allow us to complete the proof in the case that  $M_k$  is indefinite.

This first lemma has been adapted from He and Yuan [7]. It shows that the stability condition **C1** forces the matrix  $M_k$  to be positive definite. When this condition is satisfied, the operator  $\langle u, M_k u \rangle = \|u\|_{M_k}^2$  is a proper norm and can be used in our convergence analysis.

Lemma 1 also shows that the backtracking step in Algorithm 3 can only be triggered a finite number of times. When the product  $\tau_k \sigma_k$  becomes sufficiently small, so does the constant  $C_M$  in the lemma, and the ratio (26) will always be less than 1.

**Lemma 1.** *If  $\tau\sigma\|A^T A\|_{op} < 1$  then  $M$  is positive definite, and we have*

$$(1 - C_M) \left( \frac{1}{\sigma} \|y\|^2 + \frac{1}{\tau} \|x\|^2 \right) \leq \|u\|_M^2 \leq (1 + C_M) \left( \frac{1}{\sigma} \|y\|^2 + \frac{1}{\tau} \|x\|^2 \right)$$

where  $C_M = \sqrt{\tau\sigma\|A^T A\|_{op}}$ .

*Proof.* By definition

$$(35) \quad \langle u, Mu \rangle = \frac{1}{\tau} \|x\|^2 - 2\langle Ax, y \rangle + \frac{1}{\sigma} \|y\|^2.$$

Now use the inequality  $a^2 + b^2 > 2ab$  to obtain

$$(36) \quad \begin{aligned} 2\langle Ax, y \rangle &\leq 2 \frac{\|A\|_{op} \sqrt{\tau} \|y\|}{(\tau\sigma\|A^T A\|_{op})^{\frac{1}{4}}} \frac{(\tau\sigma\|A^T A\|_{op})^{\frac{1}{4}}}{\sqrt{\tau}} \|x\| \\ &\leq \frac{\|A^T A\|_{op} \tau}{\sqrt{\tau\sigma\|A^T A\|_{op}}} \|y\|^2 + \frac{\sqrt{\tau\sigma\|A^T A\|_{op}}}{\tau} \|x\|^2 \\ &= \sqrt{\tau\sigma\|A^T A\|_{op}} \left( \frac{1}{\sigma} \|y\|^2 + \frac{1}{\tau} \|x\|^2 \right). \end{aligned}$$

Applying 36 to 35 yeilds

$$\|u\|_M^2 \geq \left( 1 - \sqrt{\tau\sigma\|A^T A\|_{op}} \right) \left( \frac{1}{\sigma} \|y\|^2 + \frac{1}{\tau} \|x\|^2 \right).$$

We also have that

$$\begin{aligned} \langle u, Mu \rangle &= \frac{1}{\tau} \|x\|^2 - 2\langle Ax, y \rangle + \frac{1}{\sigma} \|y\|^2 \\ &\leq \frac{1}{\tau} \|x\|^2 + 2\|A\|_{op} \|x\| \|y\| + \frac{1}{\sigma} \|y\|^2 \\ &\leq \frac{1}{\tau} \|x\|^2 + \frac{1}{\sigma} \|y\|^2 + \sqrt{\tau\sigma\|A^T A\|_{op}} \left( \frac{1}{\sigma} \|y\|^2 + \frac{1}{\tau} \|x\|^2 \right) \end{aligned}$$

and so

$$\|u\|_M^2 \leq \left( 1 + \sqrt{\tau\sigma\|A^T A\|_{op}} \right) \left( \frac{1}{\sigma} \|y\|^2 + \frac{1}{\tau} \|x\|^2 \right).$$

□

We next show that the distance between the true solution and the PDHG iterates is decreasing in the  $M_k$  norm. If  $M_k$  were constant and positive definite, then this result would be sufficient for convergence. However, because the matrix  $M_k$  changes at each iteration and may be indefinite, this condition does not guarantee convergence on its own.

**Lemma 2.** *The iterates generated by Algorithm 1 satisfy*

$$\|u_k - u^*\|_{M_k}^2 \geq \|u_{k+1} - u_k\|_{M_k}^2 + \|u_{k+1} - u^*\|_{M_k}^2.$$

*Proof.* Subtracting (31) from (32) gives us

$$M_k(u_{k+1} - u_k) \in R(u^*) - R(u_{k+1}).$$

Taking the inner product with  $(u^* - u_{k+1})$  gives us

$$\langle u^* - u_{k+1}, M_k(u_{k+1} - u_k) \rangle \geq \langle u^* - u_{k+1}, R(u^*) - R(u_{k+1}) \rangle.$$

Because  $R$  is monotone, the right hand side of the above equation is non-negative, and so

$$(37) \quad \langle u^* - u_{k+1}, M_k(u_{k+1} - u_k) \rangle \geq 0.$$

Now, observe the identity

$$\begin{aligned} \|u_k - u^*\|_{M_k}^2 &= \|u_{k+1} - u_k\|_{M_k}^2 + \|u_{k+1} - u^*\|_{M_k}^2 \\ &\quad + 2\langle u_k - u_{k+1}, u_{k+1} - u^* \rangle_{M_k}. \end{aligned}$$

Applying (37) gives us

$$(38) \quad \|u_k - u^*\|_{M_k}^2 \geq \|u_{k+1} - u^*\|_{M_k}^2 + \|u_{k+1} - u_k\|_{M_k}^2.$$

□

We now show that the iterates generated by PDHG remain bounded.

**Lemma 3.** *Suppose the step sizes for Algorithm 1 satisfy conditions **A**, **B** and **C**. Then*

$$\|u_k - u^*\|_{H_k}^2 \leq C_U$$

for some upper bound  $C_U > 0$ .

*Proof.* We first consider the case of condition **C1**. From (36) we have

$$2\langle Ax, y \rangle \leq \sqrt{\tau\sigma\|A^T A\|_{op}} \left( \frac{1}{\sigma}\|y\|^2 + \frac{1}{\tau}\|x\|^2 \right).$$

Subtracting  $2\sqrt{\tau\sigma\|A^T A\|_{op}}\langle Ax, y \rangle$  from both sides yields

$$\begin{aligned} \left( 2 - 2\sqrt{\tau\sigma\|A^T A\|_{op}} \right) \langle Ax, y \rangle &\leq \sqrt{\tau\sigma\|A^T A\|_{op}} \left( \frac{1}{\sigma}\|y\|^2 + \frac{1}{\tau}\|x\|^2 - 2\langle Ax, y \rangle \right) \\ &= \|u\|_M^2 \sqrt{\tau\sigma\|A^T A\|_{op}}. \end{aligned}$$

Taking  $x = x_{k+1} - x^*$ , and  $y = y_{k+1} - y^*$ ,  $\tau = \tau_{k+1}$ , and  $\sigma = \sigma_{k+1}$  we obtain

$$2\langle A(x_{k+1} - x^*), y_{k+1} - y^* \rangle \leq C_1 \|u_{k+1} - u^*\|_{M_{k+1}}^2$$

where  $C_1 = \frac{\sqrt{\tau_{k+1}\sigma_{k+1}\|A^T A\|_{op}}}{1 - \sqrt{\tau_{k+1}\sigma_{k+1}\|A^T A\|_{op}}} > 0$ .

Applying this to the result of Lemma 2, yields

$$\begin{aligned} \|u_k - u^*\|_{M_k}^2 &\geq \|u_{k+1} - u^*\|_{M_k}^2 \\ &= \|u_{k+1} - u^*\|_{H_k}^2 + \langle A(x_{k+1} - x^*), y_{k+1} - y^* \rangle \\ &\geq \delta_k \|u_{k+1} - u^*\|_{H_{k+1}}^2 + \langle A(x_{k+1} - x^*), y_{k+1} - y^* \rangle \\ &\geq \delta_k \|u_{k+1} - u^*\|_{M_{k+1}}^2 + (1 - \delta_k) \langle A(x_{k+1} - x^*), y_{k+1} - y^* \rangle \\ &\geq \delta_k \|u_{k+1} - u^*\|_{M_{k+1}}^2 - (1 - \delta_k) C_1 \|u_{k+1} - u^*\|_{M_{k+1}}^2 \\ &= (1 - (1 + C_1)\phi_k) \|u_{k+1} - u^*\|_{M_{k+1}}^2. \end{aligned}$$

It follows that

$$(39) \quad \|u_1 - u^*\|_{M_1}^2 \geq \prod_{k=1}^{n-1} (1 - (1 + C_1)\phi_k) \|u_n - u^*\|_{M_n}^2.$$

Since  $\sum_k \phi_k < \infty$ , we have that  $\sum_k (1 + C_1)\phi_k < \infty$ , and the product on the right of (39) is bounded away from zero. Thus, there is a constant  $C_2$  with

$$\|u_1 - u^*\|_{M_1}^2 \geq C_2 \|u_n - u^*\|_{M_n}^2 \geq C_2 (1 - C_M) \|u_n - u^*\|_{H_n}^2$$

and the lemma is true in the case of assumption **C1**.



We now consider the case where condition **C2** holds. We assume without loss of generality that  $Y$  is bounded (the case of bounded  $X$  follows by nearly identical arguments). In this case, we have  $\|y\| \leq C_y$  for all  $y \in Y$ .

Note that

$$\begin{aligned}
 \|u_{k+1} - u^*\|_{M_k} &= \frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 - \langle A(x_{k+1} - x^*), y_{k+1} - y^* \rangle + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2 \\
 (40) \quad &\geq \frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 - 2C_y \|A\|_{op} \|x_{k+1} - x^*\| + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2.
 \end{aligned}$$

When  $\|x_{k+1} - x^*\|$  grows sufficiently large, the term involving the square of this norm dominates the value of (40). Since  $\{\tau_k\}$  and  $\{\sigma_k\}$  are bounded from above, it follows that there is some positive  $C_x$  such that

$$\begin{aligned}
 \frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2 &\geq C_x \\
 \implies \frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2 &\geq 4C_y \|A\|_{op} \|x_{k+1} - x^*\| \\
 (41) \quad \implies 2\|u_{k+1} - u^*\|_{M_k} &\geq \frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2.
 \end{aligned}$$

Whenever  $\frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2 \geq C_x$  we have

$$\begin{aligned}
 \|u_{k+1} - u^*\|_{M_k} &= \frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 - \langle A(x_{k+1} - x^*), y_{k+1} - y_k \rangle + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2 \\
 &\geq \frac{\delta_k}{\tau_{k+1}} \|x_{k+1} - x^*\|^2 - \langle A(x_{k+1} - x^*), y_{k+1} - y^* \rangle + \frac{\delta_k}{\sigma_{k+1}} \|y_{k+1} - y^*\|^2 \\
 &= \|u_{k+1} - u^*\|_{M_{k+1}} - \frac{\phi_k}{\tau_{k+1}} \|x_{k+1} - x^*\|^2 - \frac{\phi_k}{\sigma_{k+1}} \|y_{k+1} - y^*\|^2 \\
 (42) \quad &\geq (1 - 2\phi_k) \|u_{k+1} - u^*\|_{M_{k+1}}.
 \end{aligned}$$

Applying (42) to Lemma 2 yields

$$(43) \quad \|u_k - u^*\|_{M_k}^2 \geq (1 - 2\phi_k) \|u_{k+1} - u^*\|_{M_{k+1}}^2.$$

Now, consider the case that  $\frac{1}{\tau_k} \|x_{k+1} - x^*\|^2 + \frac{1}{\sigma_k} \|y_{k+1} - y^*\|^2 < C_x$ . We have

$$\begin{aligned}
 \|u_{k+1} - u^*\|_{M_k} &\geq \|u_{k+1} - u^*\|_{M_{k+1}} - \frac{\phi_k}{\tau_{k+1}} \|x_{k+1} - x^*\|^2 - \frac{\phi_k}{\sigma_{k+1}} \|y_{k+1} - y^*\|^2 \\
 (44) \quad &\geq \|u_{k+1} - u^*\|_{M_{k+1}} - \phi_k C_x.
 \end{aligned}$$

Applying (44) to Lemma 2 yields

$$(45) \quad \|u_k - u^*\|_{M_k}^2 \geq \|u_{k+1} - u^*\|_{M_{k+1}}^2 - \phi_k C_x.$$

From (43) and (45), it follows by induction that

$$(46) \quad \|u_0 - u^*\|_{M_k}^2 \geq \prod_{i \in I_C} (1 - 2\phi_i) \|u_{k+1} - u^*\|_{M_{k+1}}^2 - \sum_i \phi_i C_x$$

where  $I_C = \{i \mid \|x_{i+1} - x^*\| \geq C_x\}$ .

We can rearrange (46) to obtain

$$\|u_{k+1} - u^*\|_{M_{k+1}}^2 \leq \frac{\|u_0 - u^*\|_{M_k}^2 + C_x \sum_i \phi_i}{\prod_i (1 - 2\phi_i)} < \infty$$

which shows that  $\|u_k - u^*\|_{M_k}^2$  remains bounded (note:  $\lim_{k \rightarrow \infty} \phi_i = 0$ , and so we may assume without loss of generality that  $\{1 - 2\phi_i\}$  is positive).

Finally, note that since  $\{\tau_k\}$ ,  $\{\sigma_k\}$ , and  $\|u_k - u^*\|_{M_k}$  are bounded from above, it follows from (40) that  $\frac{1}{\tau_k}\|x_k - x^*\|^2$  is bounded from above. But  $\frac{1}{\sigma_k}\|y_k - y^*\|^2$  is also bounded from above, and so  $\|u_k - u^*\|_{H_k}$  is bounded as well.  $\square$

Lemma 3 established upper bounds on the sequence of iterates. To complete our convergence proof we also need lower bounds on  $\|u_k - u^*\|_{M_k}^2$ . Note that in the case of indefinite  $M_k$ , this quantity may be negative. The following result shows that  $\|u_k - u^*\|_{M_k}^2$  does not approach negative infinity.

**Lemma 4.** *Suppose the step sizes for Algorithm 1 satisfy conditions **A**, **B**, and **C**. Then*

$$\|u_{k+1} - u^*\|_{M_k}^2 \geq C_L$$

for some lower bound  $C_L$ .

*Proof.* In the case that **C1** holds, Lemma 1 tells us that  $M_k$  is positive definite. In this case  $\|\cdot\|_{M_k}^2$  is a proper norm and

$$\|u_{k+1} - u^*\|_{M_k}^2 \geq 0.$$

In the case that **C2** holds, we have that either  $X$  or  $Y$  is bounded. Assume without loss of generality that  $Y$  is bounded. We then have  $\|y\| < C_y$  for all  $y \in Y$ . We can then obtain

$$\begin{aligned} \|u_{k+1} - u^*\|_{M_k} &\geq \frac{1}{\tau_k}\|x_{k+1} - x^*\|^2 - 2C_y\|A\|_{op}\|x_{k+1} - x^*\| + \frac{4C_y^2}{\sigma_k} \\ (47) \quad &\geq \frac{1}{\min\{\tau_k\}}\|x_{k+1} - x^*\|^2 - 2C_y\|A\|_{op}\|x_{k+1} - x^*\| + \frac{4C_y^2}{\min\{\sigma_k\}}. \end{aligned}$$

Note that (47) is quadratic in  $\|x_{k+1} - x^*\|$ , and so this quantity is bounded from below.  $\square$

We are now ready to prove the main result. The idea is to show that the norms of the residuals have a finite sum and thus converge to zero. Because the “natural” norm of the problem, the  $M_k$ -norm, changes after each iteration, we must be careful to bound the differences between the norms used at each iteration. For this purpose, condition **B** will be useful because it guarantees that the various  $M_k$ -norms do not differ too much as  $k$  gets large.

**Theorem 1.** *Suppose that the stepsizes in Algorithm 1 satisfy conditions **A** and **B**, and either **C1** or **C2**. Then the algorithm converges in the residuals, i.e.*

$$\lim_{k \rightarrow \infty} \|P_k\|^2 + \|D_k\|^2 = 0.$$

*Proof.* Rearranging Lemma (37) gives us

$$(48) \quad \|u_k - u^*\|_{M_k}^2 - \|u_{k+1} - u^*\|_{M_k}^2 \geq \|u_{k+1} - u_k\|_{M_k}^2.$$

Summing (48) for  $1 \leq k \leq n$  gives us

$$\begin{aligned} (49) \quad &\|u_1 - u^*\|_{M_0}^2 - \|u_{n+1} - u^*\|_{M_n}^2 + \sum_{k=1}^n \|u_k - u^*\|_{M_k}^2 - \|u_k - u^*\|_{M_{k-1}}^2 \\ &\geq \sum_{k=1}^n \|u_{k+1} - u_k\|_{M_k}^2. \end{aligned}$$

Now, we expand the summation on the left side of (48) using the definition of  $M_k$  to obtain

$$\begin{aligned}
 (50) \quad & \sum_{k=1}^n \|u_k - u^*\|_{M_k}^2 - \|u_k - u^*\|_{M_{k-1}}^2 \\
 &= \sum_{k=1}^n \left( \frac{1}{\tau_k} - \frac{1}{\tau_{k-1}} \right) \|x_k - x^*\|^2 + \left( \frac{1}{\sigma_k} - \frac{1}{\sigma_{k-1}} \right) \|y_k - y^*\|^2 \\
 &\leq \sum_{k=1}^n (1 - \delta_{k-1}) \left( \frac{1}{\tau_k} \|x_k - x^*\|^2 + \frac{1}{\sigma_k} \|y_k - y^*\|^2 \right) \\
 &= \sum_{k=1}^n \phi_{k-1} \|u_k - u^*\|_{H_k}^2 \leq C_U \sum_{k=1}^n \phi_{k-1} < \infty,
 \end{aligned}$$

where we have used the bound  $\|u_k - u^*\|_{H_k}^2 < C_U$  from Lemma 3.

Applying (50) to (49), and noting from Lemma 4 that  $\|u_{n+1} - u^*\|_{M_n}$  is bounded from below, we see that

$$\sum_{k=1}^n \|u_{k+1} - u_k\|_{M_k}^2 < \infty,$$

and it follows that  $\lim_{k \rightarrow \infty} \|u_{k+1} - u_k\|_{M_k} = 0$ . If condition **C2** holds, then this clearly implies that

$$(51) \quad \lim_{k \rightarrow \infty} \|u_{k+1} - u_k\|_{H_k} = 0.$$

If condition **C1** holds, then we still have (51) from Lemma 1. Since  $\tau_k$  and  $\sigma_k$  are bounded from above, (51) implies that

$$(52) \quad \lim_{k \rightarrow \infty} \|u_{k+1} - u_k\| = 0.$$

Recall the definition of  $R_k$  in equation (29). From (31), we know that

$$R_k = M_k(u_k - u_{k+1}) \in R(u_{k+1}),$$

and so

$$(53) \quad \lim_{k \rightarrow \infty} \|R_k\| = \lim_{k \rightarrow \infty} \|M(u_{k+1} - u_k)\| \leq \max_k \{\|M_k\|\} \lim_{k \rightarrow \infty} \|u_{k+1} - u_k\| = 0.$$

□

## 7. NUMERICAL RESULTS

To demonstrate the performance of the new adaptive PDHG schemes, we apply them to the test problems described in Section 3. We run the algorithms with parameters  $\alpha_0 = 0.5$ ,  $\Delta = 1.5$ , and  $\eta = 0.95$ . The backtracking method is run with  $\gamma = .75$ ,  $\beta = .95$ , and we initialize stepsizes using formula (27). We terminate the algorithms when both the primal and dual residual norms (i.e.  $|P_k|$  and  $|D_k|$ ) are smaller than 0.05, unless otherwise specified.

We consider four variants of PDHG. The method “Adapt:Backtrack” denotes the adaptive backtracking method (Algorithm 3) with large initial stepsizes chosen according to (27). The method “Adapt:  $\tau\sigma = L$ ” refers to the adaptive method without backtracking (Algorithm 2) with  $\tau_0 = \sigma_0 = 0.95\rho(A^T A)^{-\frac{1}{2}}$ .

We also consider the non-adaptive PDHG with two different stepsize choices. The method “Const:  $\tau, \sigma = \sqrt{L}$ ” refers to the constant-stepsize method with both stepsize parameters equal to  $\sqrt{L} = \rho(A^T A)^{-\frac{1}{2}}$ . The method “Const:  $\tau$ -final” refers to the constant-stepsize

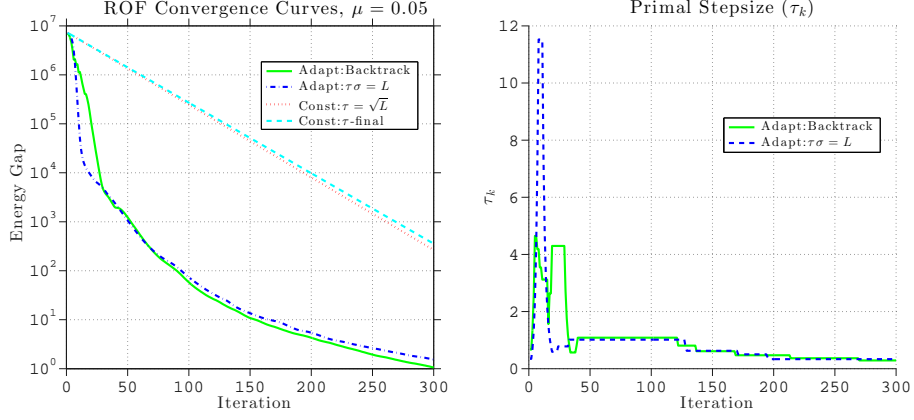


FIGURE 1. (left) Convergence curves for the Rudin-Osher-Fatemi denoising experiment with  $\mu = 0.05$ . The  $y$ -axis displays the difference between the value of the ROF objective function (12) at the  $k$ th iterate and the optimal objective value. (right) Stepsize sequences,  $\{\tau_k\}$ , for both adaptive schemes.

method, where the stepsizes are chosen to be the final values of the stepsizes used by “Adapt:  $\tau\sigma = L$ ”. This final method is meant to demonstrate the performance on PDHG with a stepsize that is customized to the problem at hand, but still non-adaptive.

**7.1. Experiments.** The specifics of each test problem are described below:

*Rudin-Osher-Fatemi Denoising.* We apply the denoising model (12) to the “Cameraman” test image. The image is scaled to have pixels in the range  $[0, 255]$ , and contaminated with Gaussian noise of standard deviation 10. The image is denoised with  $\mu = 0.25, 0.05$ , and 0.01.

We display denoised images in Figure 2. We show results of numerical time trails in Table 1. Note that the iteration count for denoising problems increases for small  $\mu$ , which results in solutions with large piecewise -constant regions. Note also the similar performance of Algorithms 2 and 3, indicating that there is no significant advantage to knowing the constant  $L = \rho(A^T A)^{-1}$ .

We plot convergence curves and show the evolution of  $\tau_k$  in Figure 1. Note that  $\tau_k$  is large for the first several iterates and then decays over time. This behavior is typical for many TV-regularized problems.

*TVL1 Denoising.* We again denoise the “Cameraman” test image, this time using the model (17), which tends to result in smoother results. The image is denoised with  $\mu = 2, 1$ , and 0.5.

We display denoised images in Figure 2, and show time trials results in Table 1. Much like in the ROF case, the iteration counts increase as denoising results get coarser (i.e. when  $\mu$  gets small.) There is no significant advantage to specifying the value of  $L = \rho(A^T A)^T$ , because the backtracking algorithm was very effective for this problem, much like in the ROF case.

*Convex Segmentation.* We apply the model (19) to a test image containing circular regions organized in a triangular pattern. By choosing different weights for the data term  $\mu$ , we

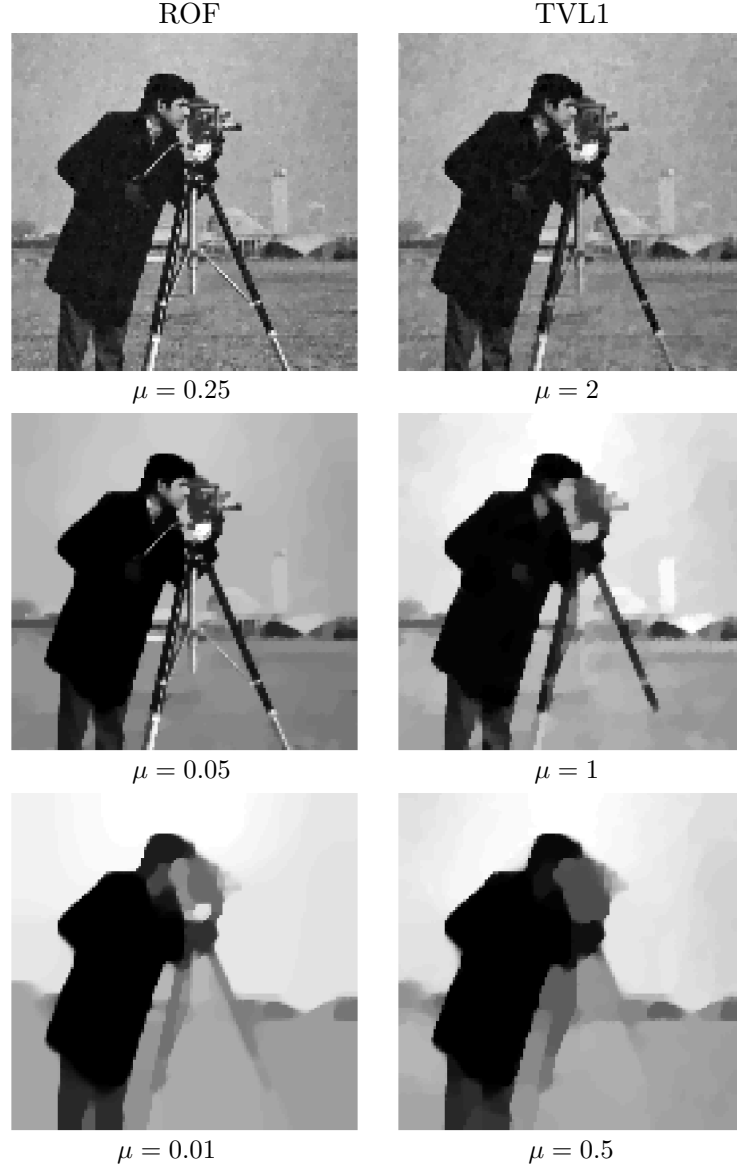


FIGURE 2. Results of denoising experiments with cameraman image. (left column) ROF results with  $\mu = 0.25$ ,  $0.05$ , and  $0.01$  from top to bottom. (right column) TVL1 results with  $\mu = 2$ ,  $1$ , and  $0.5$  from top to bottom.

achieve segmentations at different scales. In this case, we can identify each circular region as its own entity, or we can group regions together into groups of 3 or 9 circles.

Results of segmentations at different scales are presented in Figure 3. Convergence curves are shown in Figure 4 for both the energy and residuals. Note that the residuals exhibit a similar shape to the energy convergence curves, indicating that the residuals are a good measure of convergence.

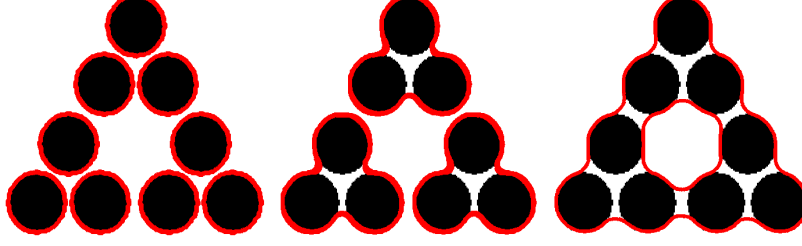
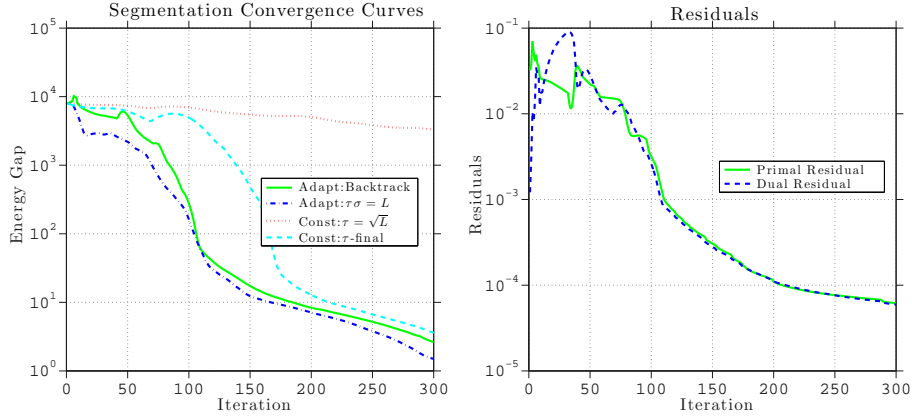


FIGURE 3. Segmentation of the “circles” test image at different scales.

FIGURE 4. (left) Convergence curves for the segmentation experiment with  $\mu = 0.15$ . (right) Primal and dual residual norms for the backtracking adaptive method.

*Compressed Sensing.* We reconstruct a Shepp-Logan phantom from sub-sampled Hadamard measurements. Data is generated by applying the Hadamard transform to a  $256 \times 256$  discretization of the Shepp-Logan phantom, and then sub-sampling 5%, 10%, and 20% of the coefficients are random. We scaled the image to have pixels in the range  $[0, 255]$ , the orthogonal scaling of the Hadamard transform was used, and we reconstruct all images with  $\mu = 1$ . The compressive reconstruction with 10% sampling is shown in Figure 5. See Table 1 for iteration counts at the various sampling rates.

*$\ell_\infty$  Minimization.* To demonstrate the performance of the adaptive schemes with complex inputs, we solve a signal approximation problem using a complex-valued tight frame. Problems of the form (22) were generated by randomly sub-sampling 100 rows from a discrete Fourier matrix of dimension 512. The input signal  $z$  was a vector on random Gaussian variables. Problems were solved with approximation accuracy  $\epsilon = 1, 0.1$ , and  $0.01$ , and results are reported in Table 1.

Because this problem is complex-valued, we use the definition of  $b_k$  given by (28) to guarantee that the stepsizes are real-valued.

*General Linear Programming.* We test our algorithm on the standard test problem “sc50b” that ships with most distributions of Matlab. The problem is to recover 40 unknowns subject

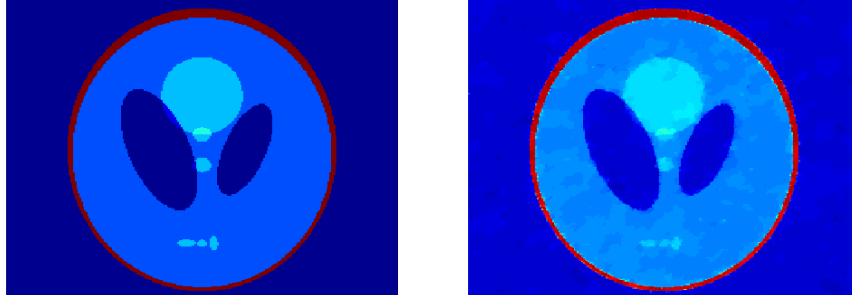


FIGURE 5. Compressed sensing reconstruction experiment. (left) Original Shepp-Logan phantom. (right) Image reconstructed from a random 10% sample of noisy Hadamard coefficients. Images are depicted in false color to accentuate small features.

to 30 inequality and 20 equality constraints. To accelerate the method, we apply a preconditioner to the problem inspired by the diagonal stepsize proposed for linear programming in [25]. This preconditioner works by replacing  $A$  and  $b$  in (23) with

$$\hat{A} = \Gamma^{\frac{1}{2}} A \Sigma^{\frac{1}{2}}, \quad \hat{b} = \Gamma^{\frac{1}{2}} b$$

where  $\Gamma$  and  $\Sigma$  are diagonal preconditioning matrices with  $\Gamma_{ii} = \sum_j |A_{i,j}|$  and  $\Sigma_{jj} = \sum_i |A_{i,j}|$ .

Time trial results are shown in Table 1. Note that PDHG is not as efficient as conventional linear programming methods for small-scale problems; the Matlab “linprog” command took approximately 0.05 seconds to solve this problem using interior point methods. Interestingly, the backtracking scheme (Algorithm 3) out-performed Algorithm 2 for this problem (see Table 1).

TABLE 1. Time Trial Results. Iteration counts are reported for each problem/algorithm, with total runtime (sec) in parenthesis.

Problem	Adapt:Backtrack	Adapt: $\tau\sigma = L$	Const: $\tau, \sigma = \sqrt{L}$	Const: $\tau$ -final
ROF, $\mu = .25$	16 (0.0475)	16 (0.041)	78 (0.184)	48 (0.121)
ROF, $\mu = .05$	50 (0.122)	51 (0.122)	281 (0.669)	97 (0.228)
ROF, $\mu = .01$	109 (0.262)	122 (0.288)	927 (2.17)	152 (0.369)
TVL1, $\mu = 2$	286 (0.957)	285 (0.954)	852 (2.84)	380 (1.27)
TVL1, $\mu = 1$	523 (1.78)	521 (1.73)	1522 (5.066)	669 (2.21)
TVL1, $\mu = .5$	846 (2.74)	925 (3.07)	3244 (10.8)	1363 (4.55)
Segment, $\mu = 0.5$	42 (0.420)	41 (0.407)	114 (1.13)	53 (0.520)
Segment, $\mu = .15$	111 (1.13)	107 (1.07)	493 (5.03)	131 (1.34)
Segment, $\mu = .08$	721 (7.30)	1016 (10.3)	706 (7.13)	882 (8.93)
Compressive (20%)	163 (4.08)	168 (4.12)	501 (12.54)	246 (6.03)
Compressive (10%)	244 (5.63)	274 (6.21)	908 (20.6)	437 (9.94)
Compressive (5%)	382 (9.54)	438 (10.7)	1505 (34.2)	435 (9.95)
$\ell_\infty$ ( $\epsilon = 1$ )	86 (0.0675)	56 (0.0266)	178 (0.0756)	38 (0.0185)
$\ell_\infty$ ( $\epsilon = .1$ )	79 (0.0360)	72 (0.0309)	195 (0.0812)	78 (0.0336)
$\ell_\infty$ ( $\epsilon = .01$ )	89 (0.0407)	90 (0.0392)	200 (0.0833)	82 (0.0364)
LP	18255 (3.98)	20926 (4.67)	24346 (5.42)	29357 (6.56)

**7.2. Discussion.** Several interesting observations can be made from the results in Table 1. First, both Algorithm 3 (“Adapt:Backtrack”) and 2 (“Adapt:  $\tau\sigma = L$ ”) have similar performance on average for the imaging problems, with neither algorithm showing consistently better performance than the other. This shows that the backtracking scheme (Algorithm 3) is highly effective and that there is no significant advantage to requiring the user to supply the constant  $\rho(A^T A)$  other than for simplicity of implementation.

Interestingly, when the backtracking method did show better performance, it was often due to the ability to use large stepsizes that violate the stepsize restriction **C1**. When solving the ROF denoising problem with  $\mu = 0.01$ , for example, the backtracking scheme terminated with  $\tau_k \sigma_k = 0.14$ , while the conventional stepsize restriction is  $1/\rho(A^T A) = 1/8 = 0.125$ .

There are two significant observations to be made from Figure 4, which depicts the convergence of the residuals for an instance of the segmentation problem. First, the residuals have the same overall shape as the energy convergence curves. This indicates that the residuals serve as a good proxy for the objective function, and are thus a reasonable measure of convergence. Second, the residuals of the initial iterates in Figure 4 differ in scale by almost two orders of magnitude. After the first few iterations of the method, the residual balancing scheme maintains that the primal and dual residuals in Figure 4 remain roughly the same size within a factor of  $\Delta = 1.5$ .

Finally, the method “Const:  $\tau$ -final” (a non-adaptive method using the “optimized” stepsizes obtained from the last iterations of the adaptive scheme) did not always outperform the non-adaptive scheme using the non-optimized stepsizes. This occurs because the true “best” stepsize choice depends on the active set of the problem in addition to the structure of the remaining error and thus evolves over time. This situation is depicted in Figure 1, which shows the evolution of  $\tau_k$  over time for the adaptive methods. Note that for this problem, both methods favor a large value for  $\tau_k$  during the first few iterations, and this value decays over time. Behavior like this is typical for the imaging problems discussed above. This illustrates the importance of an adaptive stepsize — by optimizing the stepsizes to the current active set and error levels, adaptive methods can achieve superior performance.

## 8. CONCLUSION

We have introduced new adaptive variants of the powerful Primal-Dual Hybrid Gradient (PDHG) schemes. We proved rigorous convergence results for adaptive methods and identified the necessary conditions for convergence. We also presented practical implementations of adaptive schemes that formally satisfy the convergence requirements. Numerical experiments show that adaptive PDHG methods have advantages over non-adaptive implementations in terms of both efficiency and simplicity for the user.

The new backtracking variant of PDHG is advantageous because it does not require the user to supply any information about the problem instance being solved such as the spectral radius of the matrix  $A$ . This is particularly useful when creating “black-box” solvers for large classes of problems. For example, a generic linear programming solver should not require the user to supply information about the eigenvalues of the constraint matrix. Furthermore, this flexibility seems to come at no added cost; the fully adaptive backtracking scheme performs at least as well as methods that do require spectral information.

## REFERENCES

- [1] L. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica. D.*, vol. 60, pp. 259–268, 1992.



- [2] E. Esser, X. Zhang, and T. Chan, "A general framework for a class of first order primal-dual algorithms for tv minimization," *UCLA CAM Report 09-67*, 2009.
- [3] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Convergence*, vol. 40, no. 1, pp. 1–49, 2010.
- [4] L. Popov, "A modification of the arrow-hurwicz method for search of saddle points," *Mathematical notes of the Academy of Sciences of the USSR*, vol. 28, pp. 845–848, 1980.
- [5] M. Zhu and T. Chan, "An efficient primal-dual hybrid gradient algorithm for total variation image restoration," *UCLA CAM technical report, 08-34*, 2008.
- [6] T. Pock, D. Cremers, H. Bischof, and A. Chambolle, "An algorithm for minimizing the mumford-shah functional," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 1133–1140, 2009.
- [7] B. He and X. Yuan, "Convergence analysis of primal-dual algorithms for a saddle-point problem: From contraction perspective," *SIAM J. Img. Sci.*, vol. 5, pp. 119–149, Jan. 2012.
- [8] R. T. Rockafellar, *Convex Analysis (Princeton Landmarks in Mathematics and Physics)*. Princeton University Press, Dec. 1996.
- [9] T. Chan and S. Esedoglu, "Aspects of total variation regularized  $\ell_1$  function approximation," *SIAM J. Appl. Math.*, 2005.
- [10] T. Chan, S. Esedoglu, and M. Nikolova, "Algorithms for finding global minimizers of image segmentation and denoising models," *SIAM Journal on Applied Mathematics*, vol. 66, pp. 1932–1648, 2006.
- [11] T. Goldstein, X. Bresson, and S. Osher, "Geometric applications of the split bregman method: Segmentation and surface reconstruction," *J. Sci. Comput.*, vol. 45, pp. 272–293, October 2010.
- [12] T. Goldstein, X. Bresson, and S. Osher, "Global minimization of markov random fields with applications to optical flow," *Inverse Problems in Imaging*, vol. 6, pp. 623–644, November 2012.
- [13] E. Brown, T. Chan, and X. Bresson, "Completely convex formulation of the chan-veese image segmentation model," *International Journal of Computer Vision*, vol. 98(1), pp. 103–121, 2012.
- [14] B. Goldluecke and D. Cremers, "Convex Relaxation for Multilabel Problems with Product Label Spaces," in *Computer Vision ECCV 2010* (K. Daniilidis, P. Maragos, and N. Paragios, eds.), vol. 6315 of *Lecture Notes in Computer Science*, ch. 17, pp. 225–238, Berlin, Heidelberg: Springer Berlin / Heidelberg, 2010.
- [15] E. Bae, J. Yuan, and X. C. Tai, "Global minimization for continuous multiphase partitioning problems using a dual approach," *International journal of computer vision*, vol. 92, no. 1, pp. 112–129, 2011.
- [16] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inform. Theory*, vol. 52, pp. 489 – 509, 2006.
- [17] E. J. Candes and J. Romberg, "Signal recovery from random projections," *Proceedings of SPIE Computational Imaging III*, vol. 5674, pp. 76 – 86, 2005.
- [18] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk, "Single-pixel imaging via compressive sampling [Building simpler, smaller, and less-expensive digital cameras]," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 83–91, 2008.
- [19] S. H. Han and J. H. Lee, "An overview of peak-to-average power ratio reduction techniques for multi-carrier transmission," *Wireless Communications, IEEE*, vol. 12, no. 2, pp. 56–65, 2005.
- [20] C. Studer, W. Yin, and R. G. Baraniuk, "Signal representations with minimum  $l_\infty$ -norm," in *Proc. 50th Annual Allerton Conference on Communication, Control, and Computing*, 2012.
- [21] J. A. Cadzow, "Algorithm for the minimum-effort problem," *Automatic Control, IEEE Transactions on*, vol. 16, no. 1, pp. 60–63, 1971.
- [22] H. Jégou, T. Furon, and J. J. Fuchs, "Anti-sparse coding for approximate nearest neighbor search," in *ICASSP - 37th International Conference on Acoustics, Speech, and Signal Processing*, (Kyoto, Japan), Jan. 2012. Quaero.
- [23] J. Duchi, S. S. Schwartz, Y. Singer, and T. Chandra, "Efficient projections onto the  $\ell_1$ -ball for learning in high dimensions," in *Proceedings of the 25th international conference on Machine learning*, ICML '08, (New York, NY, USA), pp. 272–279, ACM, 2008.
- [24] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2003.
- [25] T. Pock and A. Chambolle, "Diagonal preconditioning for first order primal-dual algorithms in convex optimization," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 1762–1769, 2011.
- [26] B. He, H. Yang, and S. Wang, "Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities," *Journal of Optimization Theory and Applications*, vol. 106, no. 2, pp. 337–356, 2000.

- [27] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers,” *Foundations and Trends in Machine Learning*, 2010.
- [28] H. Bauschke and P. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.